Wielu czytelników zarzuca nam zbyt małe zainteresowanie mikrokomputerem COMMODORE C64. Faktycznie, liczba jego użytkowników gwałtownie wzrasta, a wolnorynkowa cena spadła poniżej 150 tys. złotych (luty 1985)... Zniesienie ograniczeń celnych zwiększyło możliwości zakupu mikrokomputera i wiele osób stoi przed dylematem: co wybrać. Zamieszczamy więc tym razem porównanie niektórych cech C64 z ZX SPECTRUM — z pozycji tzw. przeciętnego użytkownika, którego mniej interesuje, co siedzi w skrzynce, a bardziej — co z tego wynika. Nieco przewrotnie pominęliśmy możliwość wykorzystywania obu komputerów jako uniwersalnych „maszynek" do gier telewizyjnych, choć jest to niestety bardzo częsta motywacja zakupu. Może jednak uda nam się przekonać czytelników, że komputery powinny przede wszystkim p r a c o w a ć, czyli — ułatwiać nam życie. A to, że równocześnie mogą nam dostarczać rozrywki, jest jedynie ich cechą dodatkową, drugorzędną.

Rzetelnie zaznaczamy, że zarówno ZX SPECTRUM jak i C64 są już w wieku przedemerytalnym. Niemniej o użyteczności komputera w znacznie większym stopniu decyduje ilość dostępnego oprogramowania, niż wewnętrzne cechy konstrukcyjne. Gotowe programy dla tych komputerów (wymieniane przez krajowych użytkowników) liczy się w setkach. Inne komputery, często o znacznie lepszych cechach funkcjonalnych, nie mają niestety w naszym kraju takiego wsparcia.

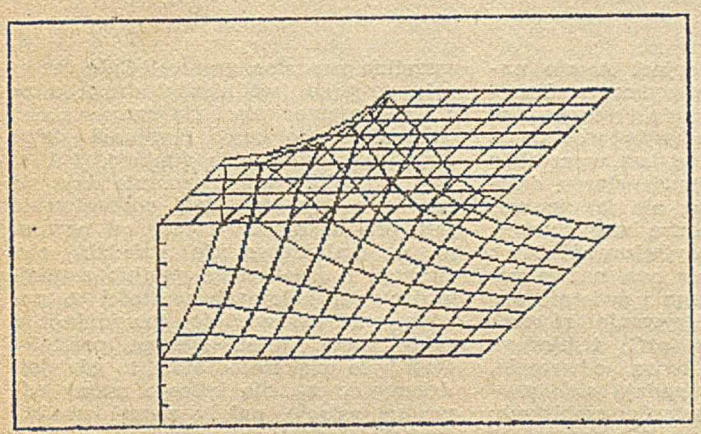
Autor poniższego tekstu wykorzystuje mikrokomputer do obliczeń naukowych. Przez pewien czas użytkował ZX SPECTRUM, a potem „przesiadł" się na C64. Oto jego refleksje.

COMMODORE C 64 czy ZX SPECTRUM?

ZX SPECTRUM, z jego możliwościami graficznymi, pozwala na niemal całkowite odejście od wyników liczbowych i prezentowanie rezultatów w formie graficznej. Problem pojawia się, gdy trzeba je jakoś utrwalić. Standardowa drukarka — to niska jakość wykresów; zdjęcia z ekranu — to kłopot.

Firma COMMODORE oferuje w stosunkowo niskiej cenie system mikrokomputerowy zawierający mikrokomputer C64, stację dysków elastycznych (!) i drukarkę lub kolorowy ploter. Jest to propozycja bez wątpienia atrakcyjna dla tych, którzy mają za mało czasu (lub wiedzy), by kompletować urządzenia różnych producentów.

Pierwsze zetknięcie z C64 było dla mnie przykrym przeżyciem. Przyzwyczajony do przemyślanego zestawu instrukcji języka BASIC w ZX SPECTRUM, z jego pełnymi możliwościami graficznymi, łatwością przechowywania na taśmie tablic z danymi lub graficzną zawartością ekranu, z prostotą przesyłania grafiki na drukarkę — byłem C64 bardzo rozczarowany. Reklamowana wysokorozdzielcza grafiki (200×320 punktów) jest trudno dostępna, przekazywanie danych poprzez instrukcję Print mało efektywne (wolne i mała gęstość zapisu), nie mówiąc już o wyprowadzaniu grafiki na drukarkę.



Przykład zobrazowania funkcji przestrzennej z wykorzystaniem C64 i drukarki MPS 801

Eksploatując już ponad rok zestaw składający się z mikrokomputera C64, stacji dyskowej 1541 i drukarki mozaikowej MPS 801, dorobiłem się zbioru kilkuset programów, sterty literatury i odrobiny doświadczeń. Od tamtej strasznej chwili pierwszego zetknięcia się z C64 zmieniłem o nim zdanie. Porównując tylko komputery — wybrałbym ZX SPECTRUM, ale przy porównaniu systemów sytuacja wygląda inaczej. Podstawową zaletą zestawu z C64 jest bowiem jego stacja dyskowa.

BASIC C64 jest bardzo ubogi. Istnieje natomiast dość bogate oprogramowanie dodatkowe, umożliwiające wykorzystanie potencjalnych możliwości C64. Aby w pełni wykorzystać możliwości graficzne, konieczne jest użycie jednego z rozszerzeń języka BASIC, np.: SIMON'S BASIC, SUPERGRAFIKA lub ULTRABASIC (dostępne instrukcje typu „postaw punkt", „wykreśl linię", „wyprowadź obraz z ekranu na drukarkę"). Dopiero teraz możliwości graficzne ZX SPECTRUM i C64 są porównywalne. SPECTRUM oferuje do dyspozycji programisty 40 KB RAM-u, a C64 — 30 KB (C64 ma potencjalnie większą pamięć — 64 KB, ale jest ona dostępna tylko w języku maszynowym).

Szybkość realizacji programów w języku BASIC dla C64 jest prawie dwukrotnie większa niż w przypadku SPECTRUM. Można ją zwiększyć (2—3-krotnie, czasem więcej), stosując kompilatory. Na C64 mogą być stosowane: AUSTRO-COMPILER, PETSPEED, DTL-COMPILER. Niestety, nie dają sobie one rady z grafiką.

Porównując pamięci masowe: magnetofony, microdrive i stację dyskową, należy podkreślić, że ZX SPECTRUM współpracuje z dowolnym typem magnetofonu (czasami są duże kłopoty z zapewnieniem poprawnej współpracy) i robi to szybko, natomiast C64 współpracuje tylko ze specjalnym, robi to wolniej, ale za to — bardzo pewnie. Istnieje program (TURBO TAPE) radykalnie zwiększający szybkość współpracy C64 z magnetofonem (10-krotnie).

Mikrodrive'y są bardzo szybkie, ale wymagają specjalnego interfejsu i okazują się czasem zawodne. Stacja dyskowa 1541, współpracująca z C64, jest bardzo wygodna, cicha, pewna, ale b a r d z o wolna. Program o wielkości 23 KB ładowany jest 60 sekund! Sytuację częściowo ratują programy szybkiego ładowania zmniejszające ten czas 4—10 krotnie.

W ciągu rocznej eksploatacji zestawu C64, stacji 1541 i drukarki MPS 801 wystąpiły pewne niesprawności. Dwukrotnie mikrokomputer po włączeniu nie zgłaszał systemu. Pozostawiony na kilka godzin bez aktywnego programu — „pada", natomiast wykonując program może pracować wiele godzin. Stacja dysków nie wykazywała żadnych niesprawności, poza koniecznością powtórnego formatowania niektórych dysków. Używam dyskietek jednostronnych — dwustronnie. Przy 50 dyskietkach różnych firm nie miałem z tego powodu żadnych kłopotów.

Po jakimś czasie w drukarce wysycha zbiorniczek tuszu i wydruki stają się blade. Nadal jednak są czytelne.

Niemal wszystkie programy pojawiające się w Europie są dostępne w kraju. Na razie nie istnieje w Polsce rynek programów na C64. Są one wymieniane lub przekazywane bezpłatnie. Liczba programów sięga tysiąca. Wśród firmowych programów można znaleźć gry (zręcznościowe, logiczne, strategiczne), programy do obliczeń finansowych (**Multiplan, Omnicalc**), do grafiki (**Supergrafika**), programy wspo-

magające programowanie — monitory i asemblery (**Supermonitor, Macrofire, Develop 64, Profi-mon**), bazy danych (**Datamat, Database, Data Manager**). Programy kopiujące (**Supercopy, F-16**) i języki (**FORTH, LOGO, PILOT, COMAL, PASCAL**, rozszerzenia języka BASIC — **Simon's Basic, UltraBasic, C64Plus, C64 exbasic, Exbasic, levell II**) Są programy służące do tworzenia grafiki i „malarstwa komputerowego" (**KoalaPainter, Moodle Doodle, Paintpic**). Istnieją także programy do redagowania tekstów (**Texter, Heswriter, Wordpro 3 plus**).

Interesującym programem jest syntetyzator mowy (SAM/reciter), umożliwiający zaprogramowanie w języku BASIC tekstu zrozumiale mówionego przez komputer (tekst angielski jest wyraźny; z polskim jest znacznie gorzej).

Kopiowanie programów napisanych przed 1984 rokiem nie przedstawia większych trudności. Najnowsze programy i te sprowadzane z USA mają nowe systemy zabezpieczeń i w zasadzie nie dają się kopiować.

Po kraju krąży trochę książek poświęconych C64, a w zasadzie — jego ukrytym możliwościom. Są to m.in.: **Programer's Reference Guide, More on the Sixty Four, Inside the Commodore 64, How to program your Commodore 64 in 6502/20 machine language, Das Grafikbuch zum Commodore 64, Beherschen Sie Ihren Commodore 64, 64 Tips & Trics, Commodore intern, Der Commodore 64 und der Rest der Welt, Das Gross Floppy Buch.** Pojawiają się też czasopisma poświęcone tylko COMMODOROWI: **Ahoy! Commodore Users, Compute, Compute's! Gazette, Power Play,** Czasopisma te zawierają wiele interesujących materiałów o użytkowaniu C64, a także — sporą gamę programów. Szczególnie interesujące są rubryki z odpowiedziami na pytania czytelników.

C64 ma dwie możliwości rozbudowy. Poprzez złącza użytkownika (ang. user port) można dołączyć płytkę umożliwiającą szybką współpracę z magnetofonem, zrealizować wyjście na drukarkę (typu Centronics), dołączyć konwertery napięć (aby uzyskać kanał szeregowy, zgodny ze standardem RS 232C) lub przetwornik analogowy. Drugą możliwość daje złącze rozszerzające (ang. expansion port), w które wkłada się moduły z ROM-ami.

Interesującą możliwością rozbudowy jest dołączenie przystawki z mikroprocesorem Z80, co umożliwia pracę w systemie CP/M. System ten jest w kraju dość popularny (możliwa jest więc wymiana — przenoszenie oprogramowania, a czasem nawet plików danych). W systemie CP/M dostępne są niezłe kompilatory języków PASCAL i FORTRAN, co pozwala na realizację poważniejszych programów i korzystanie z bogatego oprogramowania w FORTRANIE. Nadal jednak pozostaje nie rozwiązana sprawa grafiki.

Wydaje się, że C64 jest niezłym komputerem dla zaawansowanych programistów, znających język maszynowy i konstrukcję wewnętrzną komputera. Jego możliwości są w wielu przypadkach niedostępne dla przeciętnego użytkownika. Jest to też zarazem komputer dla użytkowników, którzy potrafią tylko naciskać kilka klawiszy, potrzebnych do załadowania gotowego programu.

Użytkownicy tacy jak ja, nie należący do żadnej z wymienionych grup, mają kłopoty ze zrealizowaniem swoich zamierzeń. Pocieszające jest to, że pojawiają się wciąż nowe programy, w których kolejne problemy są choć częściowo rozwiązywane. Przykładowo PASCAL daje możliwość programowania strukturalnego, ale nie ma grafiki. SIMON'S BASIC ma grafikę, ale jest wolny. Natomiast najnowsza wersja języka COMAL dla C64, zawierająca elementy PASCALA, MODULI-2, PL/1, LOGO i języka C, chyba rzeczywiście grupuje sensowny zestaw instrukcji.

Myślę, że mimo pojawienia się nowych modeli mikrokomputerów, C64 — ze względu na niewysoką cenę, potencjalne (ciągle jeszcze nie w pełni wykorzystane) możliwości a przede wszystkim ogromną ilość dostępnego oprogramowania — jeszcze długo będzie się cieszył powodzeniem.

JANUSZ PIECHNA
Warszawa

COMMODORE C64

Mikroprocesor: 6510 (kompatybilny z 6502)
Częstotliwość zegara: 1 MHz
ROM: 20 KB
RAM: 64 KB
RAM użytkowy: 38 KB (BASIC); 52 KB (kod maszynowy)
Szybkość zapisu na kasecie magnetofonowej: 600 bitów/s
Klawiatura: QWERTY (semiprofesjonalna)
Klawisze funkcyjne: 4 (podwójnie definiowane)
Klawiatura numeryczna: brak
Gnerowanie dźwięków: 3 kanały po 8 oktaw
Generator szumów: jest
Znaków w linii: 40
Liczba linii na ekranie: 25
Rozdzielczość: 320×200
Liczba kolorów: 16
Wyjście wizyjne: FBAS
Wyjście TV: PAL
Interfejs z drukarką: Port równoległy + nakładka programowa dla drukarek z wejściem Centronics
Interfejs szeregowy: funkcjonalnie V24 (elektrycznie wymaga przystawki)
Standardowy język: MICROSOFT BASIC
Rozszerzenia: dyski elastyczne +CP/M (zapis dyskietek nie kompatybilny); manipulatory; pióro świetlne
Cena: 700 marek RFN — 190 funtów ang.

Akademia uczy eleganckiego programowania. Przeznaczona jest dla osób mających podstawowe doświadczenie w programowaniu. Zainteresowanych prosimy o skomentowanie dotychczasowych „wykładów".

Akademia mikroKLANU (6)

Dekada

W tym odcinku odejdziemy nieco od strukturalizacji programów — prezentowana procedura zupełnie nie nadaje się do włączenia w długi program, choć jest procedurą iteracyjną. A nie nadaje się, gdyż chcemy wykorzystać najlepszy komputer — ludzki mózg — w algorytmie heurystycznym. Działanie heurystyczne jest wprawdzie absolutnym przeciwstawieństwem działania algorytmicznego, ale jak inaczej można nazwać procedurę heurystyczną, która daje się zalgorytmizować? Nasz algorytm heurystyczny nosi nazwę **poszukiwania dekadowego** i świetnie nadaje się do rozwiązywania równań przestępnych metodą prób i błędów.

Chociaż z reguły bywa odwrotnie, omawiany algorytm nadaje się wyłącznie dla komputerów osobistych: niechby ktoś spróbował pracować w ten sposób w systemie interakcyjnym. Szybko koszt jego pracy przekroczyłby wszelkie rozsądne granice! Całe szczęście, że prąd jest niezbyt drogi, a za mikrokomputer płaci się raz.

Problem dotyczy równania typu $f(x)=g(x)$, gdzie obie funkcje, f i g, są „skomplikowane", tzn. nie dają się łatwo odwracać. Idea poszukiwania jest prosta: należy zacząć od pewnego x_0 (należącego do dziedzin obu funkcji) i zwiększając go (lub zmniejszając) dla kolejnych wartości x_i porównywać, czy wartości obu funkcji są takie same. Jeżeli np. na początku $f(x_0)$ było większe od $g(x_0)$, to gdy dojdziemy do x_n, dla którego mamy odwrotną relację, należy zacząć zmniejszać (lub zwiększać) argument o krok dziesięciokrotnie mniejszy, przy czym decyzję pozostawiamy tu użytkowniko-

wi. Niewielka liczba kroków wystarczy by osiągnąć rozwiązanie równania z dużą dokładnością, a program, którego używamy, liczy tylko kilkanaście linijek. Porównajmy to z bibliotecznymi programami służącymi do rozwiązywania równań przestępnych. Prawda, jaka potęga tkwi w naszym komputerze osobistym?

Proponujemy Czytelnikom rozwiązanie przykładu przy $x_0=1$ oraz $dx=1$. Mamy nadzieję, że wszystkim uczestnikom Akademii udało się osiągnąć rozwiązanie (1,5132...). Proszę zwrócić uwagę, że nasz program jest stabilny, tzn. trudno go wyprowadzić z równowagi, a jednocześnie pozwala znajdować wiele różnych rozwiązań. Wszystko w rękach użytkownika.

W ten sposób nasza Akademia doszła do najpoważniejszego problemu eleganckiego programowania — algorytmów. Tak naprawdę dobre, efektywne programowanie to właśnie znajdowanie dobrych algorytmów, cała reszta jest już tylko kodowaniem. Naszym Czytelnikom polecamy liczne książki, wydane przez WNT w serii INFORMATYKA, gdzie można znaleźć algorytmy wielu interesujących problemów. W szczególności — książkę „Zbiór zdań z informatyki" F. L. Bauera, R. Gnatza i U. Hilla (WNT, 1981).

JAKUB TATARKIEWICZ

Oto nasza procedura:

```
LIST

5 REM PROCEDURA POSZUKIWANIA DEKADOWEGO,
6 REM WSTAW FUNKCJE W LINIACH 10 ORAZ 20.
10 DEF FN F(X)=X*COS(X)
20 DEF FN G(X)=SIN(X)/(X+10)
30 INPUT "PUNKT STARTOWY X0=";X0
40 INPUT "KROK POCZATKOWY DX=";DX
50 X=X0-DX
60 X=X+DX
70 Y1=FN F(X)
80 Y2=FN G(X)
90 PRINT "X=";X,"F=";Y1,"G=";Y2
100 INPUT "CZY ZMNIEJSZYC KROK (1-TAK,0-NIE)";P
110 IF P=0 THEN GOTO 60
120 DX=-DX/10
130 PRINT "NOWY KROK DX=";ABS(DX)
140 INPUT "MOZE JUZ WYSTARCZY (1-TAK,0-NIE)";P
150 IF P=0 THEN  GOTO 60
160 PRINT "ROZWIAZANIE X=";X
170 END
READY.
```

Po drugim odcinku Akademii pt. „Silnia" otrzymałem sygnały, że program nie działa. Rzeczywiście, pewne komputery nie dają tego samego wyniku gdy mnożymy przez 0,1 w porównaniu z dzieleniem przez 10 (linie 100 i 140 programu).

Wszystkim kursantom polecam sprawdzenie, jak zachowuje się ich komputer — ostatecznie trzeba znać swoje narzędzie pracy.

Otwieramy nową rubrykę. Publikować będziemy w niej drobne pomysły, które — zdaniem autorów — nie są godne całego artykułu. Ale czasem na drobnych pomysłach opiera się rewelacyjna całość. Pomysły mogą dotyczyć spraw programowych, sprzętowych lub łączyć elementy obydwu. Powinny być sprawdzone (o ile nie są oczywiste) — jest to bardzo ważne, gdyż redakcja nie ma na razie takich możliwości. Autor pomysłu firmuje go więc swoim nazwiskiem. Każdy opublikowany pomysł premiowany będzie egzemplarzem INFORMATYKI i nagrodą w wysokości od 500 do 1000 złotych. O wysokości nagrody decyduje, zgodnie ze swoim sumieniem i poglądami, redaktor mikroKLANU. Decyzja jest ostateczna, choć podobno można ją zaskarżyć do sądu.

Ponieważ mikroKLAN przygotowywany jest z paromiesięcznym wyprzedzeniem, w kilku najbliższych numerach — oczekując na odzew czytelników — publikować będziemy pomysły zaczerpnięte z różnych pism zachodnich.

I jeszcze jedna uwaga. Nadsyłane pomysły mogą być inspirowane przez materiały ukazujące się w innych pismach. Powinny jednak zawierać przynajmniej pewien wkład wniesiony przez nadsyłającego. Prosimy w takich przypadkach o podawanie źródła inspiracji.

POMYSŁ!

Jak odzyskać kontrolę nad programem realizowanym z pamięci, której nie ma?

W nowszych konstrukcjach mikrokomputerów 8-bitowych zaczyna brakować przestrzeni adresowej i stosuje się różne sztuczki, aby dołączyć trochę dodatkowego RAM-u lub ROM-u oprócz przepisowych 64 KB. Nadal jednak istnieją systemy z „dziurami adresowymi", w których nie ma żadnej pamięci (np. ZX SPECTRUM — wersja 16K). Dziury zdarzają się zresztą nawet w systemach z przełączanymi blokami pamięci.

Skoro jest dziura, to zgodnie z prawami Murphy'ego prędzej czy później w nią wpadniemy. Może się to stać za sprawą błędnie wyliczonego adresu (dla 8080 istnieje niebezpieczny rozkaz PCHL), błędnej operacji na stosie lub... kolegi, który włączył młynek do kawy i wygenerował w ten sposób falę impulsów zakłócających.

Istnieje jednak pewien sposób na wydobycie się z takiej opresji. Otóż odczytując informację spod adresu, do którego nie jest przypisana pamięć, zazwyczaj otrzymujemy kod FF_H. Dla mikroprocesorów takich jak 8080, Z80 czy NSC800 jest to kod rozkazu RST7, powodującego wywołanie podprogramu rozpoczynającego się od adresu 0038_H. Jeśli pod adresem 0038_H umieścimy odwołanie do procedury obsługi zaistniałego błędu, to odzyskamy kontrolę nad pracą systemu. Dodatkowo, gdy wskaźnik stosu ciągle jeszcze odwołuje się do pamięci RAM, to na stosie mamy zapamiętany błędny adres, który próbował odczytać procesor. Informacja ta może ułatwić odnalezienie błędu w programie, który był wcześniej realizowany.

Rozszerzeniem opisanego pomysłu jest wypełnianie pamięci RAM kodem FF_H przed wpisaniem programu przeznaczonego do realizacji. W ten sposób, jeśli program „wyskoczy" poza wykorzystywany obszar, otrzymamy reakcję analogiczną do wcześniej opisanej.

Warto jeszcze wspomnieć o przypadkach kiedy odwołanie do nieistniejącej pamięci nie musi oznaczać odczytania kodu FF_H. Jeżeli mamy do czynienia z systemem zbudowanym na kilku płytkach, to na pojemnościach pasożytniczych — związanych z rozbudowaną szyną danych — przez pewien czas utrzymuje się informacja związana z poprzednią operacją. Gdy szyna nie zostanie wysterowana nową informacją, to wprowadzony będzie kod odpowiadający „resztkom" poprzedniej operacji. Może się więc tak niefortunnie złożyć, że odczytany kod spowoduje ponowne wejście w obszar istniejącej pamięci i realizację różnych dziwnych rzeczy. W tej sytuacji pozostaje jednak nadzieja (poparta stosunkowo dużym prawdopodobieństwem), że odczytany kod będzie inny od kodów odpowiadających rozkazom skoku (wywołanie podprogramu) i po kilku krokach pojemności rozładują się.

Niektóre standardy magistrali systemowej (np. MULTIBUS) przewidują stosowanie buforów odwracających (są one szybsze). W takim przypadku mikroprocesor odczytywałby kod 00 odpowiadający rozkazowi NOP (nic nie rób), co oczywiście eliminuje możliwość wykorzystania opisanego pomysłu. W standardach tych przewidziano jednak zabezpieczenia realizowane sprzętowo, które blokują dostęp do pustych obszarów.

AJP

Inspiracją był tekst „Software scheme restarts your stalled µP", zamieszczony w EDN, 27 grudnia 1984

Dla wielu osób w naszym kraju możliwość generowania dźwięków przez mikrokomputer to cecha mało istotna. Faktycznie gdyby wykorzystywać komputery osobiste wyłącznie do wykonywania obliczeń, to jedynym sensownym dźwiękiem mogłoby być sapanie. Stworzono je jednak do zupełnie innych celów i dlatego praktycznie każdy oferowany na Zachodzie model posiada rozbudowane możliwości generowania różnych dźwięków. Niestety, cecha ta pobudza jedynie wyobraźnię autorów gier, mimo że mogłaby chyba z równym powodzeniem urozmaicać programy użytkowe. Wydaje się, że np. grający procesor tekstu zrobiłby furorę. Zresztą pomysł odciążenia wzroku i przerzucenia niektórych zadań na inne zmysły (np. słuch) nie jest nowy. Na razie jednak nie doczekał się żadnej spektakularnej realizacji. Osoby z inwencją nadal mają szansę.

Układ firmy GENERAL INSTRUMENTS stał się niemal standardem i można go spotkać w wielu bardzo różnych mikrokomputerach (np. komputery w standardzie MSX czy AMSTRAD CPC 464).

Poniższy opis powinien okazać się użyteczny zarówno dla tych, którzy chcą rozbudować posiadany komputer, jak i tych, którzy korzystają z komputera już wyposażonego we wspomniany układ, lecz mają skąpe informacje o jego sposobie wykorzystania.

Programowany generator dźwięków AY-3-8910/12

Produkowany przez firmę GENRAL INSTRUMENTS układ programowanego generatora sygnałów akustycznych umożliwia realizację różnorodnych efektów dźwiękowych. Rola mikroprocesora sprowadza się do wpisania pewnej sekwencji programującej, po czym układ w sposób autonomiczny realizuje otrzymane polecenia.

Producent przewidział trzy niezależne od siebie wyjścia analogowe. Sprzyja to uzyskiwaniu bardziej realistycznych efektów. Pasmo możliwych do uzyskania częstotliwości rozpoczyna się na kilkunastu Hz, a kończy się powyżej 60 kHz. Częstotliwości mogą być generowane z rozdzielczością od 15 do 30 Hz (zależy to od częstotliwości zegara doprowadzonego do wejścia CLOCK). Napięcie na wyjściach analogowych może zmieniać się z dynamiką 60 dB. Poziom sygnału wyjściowego sterowany jest niezależnie dla każdego kanału przez 4-bitowy przetwornik cyfrowo-analogowy. Pozwala to uzyskać 16 poziomów głośności, które — aby lepiej dopasować się do możliwości percepcyjnych człowieka — rozłożone zostały w skali logarytmicznej. Napięcie na wyjściach analogowych zmienia się w zakresie 0—1 V.

Oprócz wyjść analogowych układ ma również uniwersalne we-wy cyfrowe. W zależności od wersji układu, przewidziany jest jeden (AY-3-8912) lub dwa (AY-3-8910) uniwersalne porty 8-bitowe. Są to jednak wyjścia o niewielkiej obciążalności (1 bramka TTL).

Opis wyprowadzeń

DA7...DA0 — 8-bitowa szyna danych wykorzystywana do komunikacji z systemem mikroprocesorowym.

A8, A9/ — linie wyboru, które mogą być wykorzystane przy zapisie adresu rejestru. Jeżeli linie nie są podłączone, układ zachowuje się tak, jakby podane były na nie stany aktywne. Jednak dla uniknięcia skutków ewentualnych zakłóceń wskazane jest połączenie A8 i A9/ odpowiednio z +5 V i masą.

RESET/ — wejście umożliwiające wyzerowanie wszystkich rejestrów przez podanie poziomu niskiego.

CLOCK — wejście zegara (1—2 MHz) wykorzystywanego do generacji przebiegów okresowych.

BDIR, BC1, BC2 — linie sterujące kierunkiem komunikacji z mikroprocesorem i rodzajem wykonywanej operacji:

BDIR	BC2	BC1	Realizowana funkcja
0	0	0	układ nie komunikuje się z mikroprocesorem
0	0	1	zapis adresu rejestru
0	1	0	układ nie komunikuje się z mikroprocesorem
0	1	1	odczyt z zaadresowanego wcześniej rejestru
1	0	0	zapis adresu rejestru
1	0	1	układ nie komunikuje się z mikroprocesorem
1	1	0	zapis do zaadresowanego wcześniej rejestru
1	1	1	zapis adresu rejestru

Niektóre funkcje można zrealizować kilkoma sposobami, co pozwala uprościć układ adaptujący sygnały generowane przez mikroprocesor.

Kanał analogowy A, B, C — wyjścia sygnałów akustycznych.

IOA7...IOA0, IOB7...IOB0 — uniwersalne porty we-wy. Jeżeli zostały zaprogramowane jako wejście, to w przypadku braku zewnętrznego sterowania odczytywane są logiczne jedynki (zastosowano wewnętrzne rezystory podciągające).

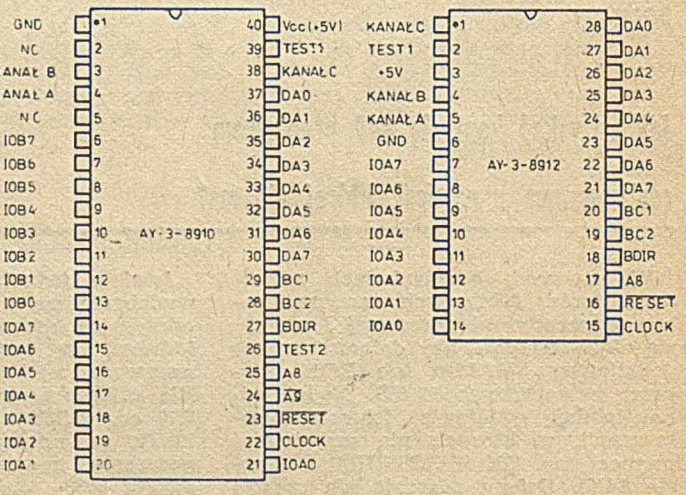
TEST 1, TEST 2 — linie wykorzystywane w procesie produkcyjnym; powinny pozostawać nie podłączone.

V_{cc} — napięcie zasilające +5 V,

V_{ss} — masa układu.

Architektura układu

Układ zawiera następujące bloki funkcjonalne:
— generator tonu podstawowego
— generator szumu
— mikser
— sterownik poziomu wyjściowego
— generator obwiedni
— przetwornik cyfrowo-analogowy
— matryca rejestrów
— porty we-wy

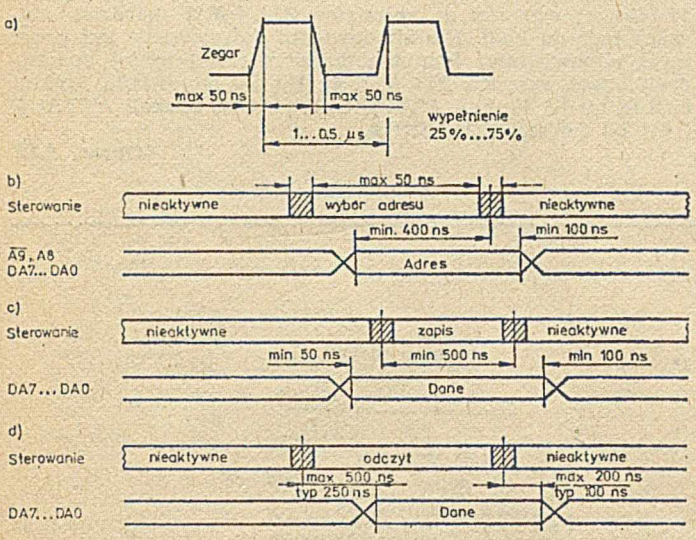


Rys. 1. Rozmieszczenie końcówek programowanego generatora dźwięków

Generator tonu podstawowego zawiera trzy niezależnie programowane bloki produkujące falę prostokątną dla każdego z kanałów. Generator szumu moduluje (częstotliwościowo) w sposób pseudolosowy przebieg prostokątny o zaprogramowanej częstotliwości bazowej.

Przebieg generatora tonu podstawowego uzyskiwany jest w wyniku podziału przez 16 sygnału zegarowego, doprowa-

dzonego do wejścia CLOCK i dodatkowego podziału przez zaprogramowaną wartość 12-bitową. Przebieg bazowy generatora szumów uzyskiwany jest w wyniku podziału sygnału zegarowego przez 16 a następnie — przez zaprogramowaną wartość 5-bitową.



Rys. 2. Przebiegi czasowe; a) zegar, b) wpisywanie adresu rejestru (na wejścia BDIR, BC2, BC1 powinna być podana jedna z sekwencji: 001, 100 lub 111), c) wpisywanie danej do zaadresowanego rejestru (na wejściach BDIR, BC2, BC1 — wymuszona sekwencja 110), d) odczytywanie danej z zaadresowanego rejestru (na wejściach BDIR, BC2, BC1 — wymuszona sekwencja 001)

Mikser umożliwia — osobno dla każdego kanału — zmieszanie szumu z przebiegiem z generatora tonu podstawowego lub wybranie jednego z tych źródeł.

Sterownik poziomu wyjściowego ustawia stały (zaprogramowany przez mikroprocesor) poziom sygnałów wychodzących z miksera lub zmienia je zależnie od chwilowych „dyspozycji" generatora obwiedni. Przetworniki C/A umożliwiają konwersję wpisanej przez mikroprocesor binarnej informacji o poziomie wyjściowym na napięcie odniesienia dla sterownika poziomu wyjściowego.

Okres, z jakim powtarzany jest kształt obwiedni, otrzymywany jest w wyniku podziału sygnału zegarowego przez 256, a następnie — przez zaprogramowaną wartość 16-bitową.

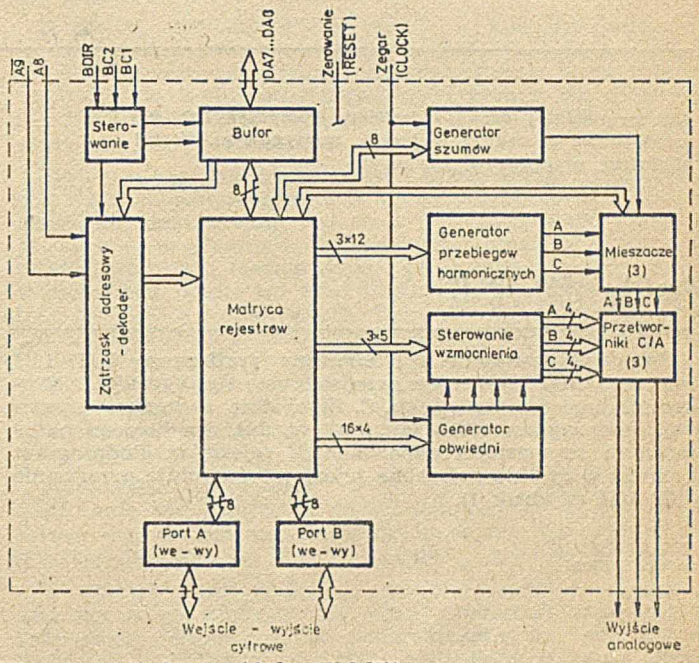
Matryca rejestrów służy do zapamiętania wpisywanych przez mikroprocesor kodów sterujących poszczególnymi blokami funkcjonalnymi układu. Dla uproszczenia programu obsługi, informacja wpisana do każdego z 16 rejestrów sterujących (R0...R15) może być w dowolnym momencie odczytana.

Rejestry R14 i R15 związane są z przesyłaniem informacji przez porty we-wy (odpowiednio: IOA i IOB). Jeżeli port został zaprogramowany jako wejście, to wprowadzaną informację można odczytać z odpowiadającego mu rejestru. Podobnie dla portu zaprogramowanego jako wyjście — informację wpisuje się do związanego z nim rejestru. Odczyt i zapis portów nie ma wpływu na przebiegi akustyczne generowane przez układ. Porty muszą być definiowane w całości (8 bitów) jako wejście lub wyjście.

Programowanie układu

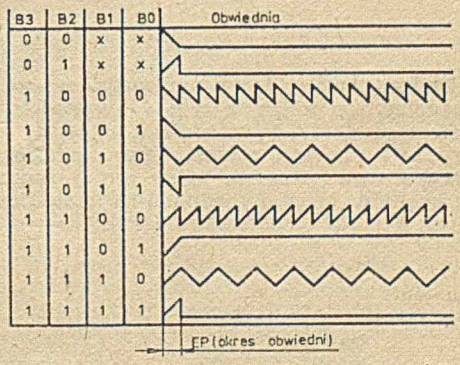
Dostęp do określonego rejestru możliwy jest po uprzednim wpisaniu adresu do układu. Adres pamiętany jest do momentu, kiedy wykonany zostanie zapis innego adresu. Umożliwia to wielokrotny dostęp do rejestru.

Wpis adresu może odbywać się dwoma sposobami, w zależności od rodzaju wykorzystywanego mikroprocesora. Dla mikroprocesorów z multipleksowaną szyną danych i adresową (na tych samych liniach podawane są na przemian dane i adresy), można zrezygnować z dodatkowego układu dekodera przestrzeni adresowej. Jeżeli na wejście BDIR, BC1, BC2, A9/, A8 podana zostanie odpowiednia kombinacja stanów logicznych, to informacja wymuszona na szynie w cyklu adresowym zostanie wprowadzona do układu. Jako ważny adres rejestru zostanie ona potraktowana tylko wtedy, jeżeli cztery bardziej znaczące bity będą zerami [1]. Cztery mniej znaczące bity odpowiadają numerowi rejestru.



Rys. 3. Architektura układu AY-3-8910

W przypadku mikroprocesorów z rozdzieloną szyną danych i adresową (np. Z80, 8080), niezbędne jest wykonanie dwóch operacji. W pierwszej zapisywana jest dana określająca rejestr, natomiast druga jest już właściwym zapisem lub odczytem żądanego rejestru.



Rys. 4. Różne typy obwiedni generowane w zależności od sekwencji wpisanej do rejestru R15

[1]) Przy zamawianiu większej liczby układów u producenta można wybrać dowolną kombinację binarną

Częstotliwość generatora tonu podstawowego programowana jest wpisem 12-bitowej wartości, przy czym cztery bardziej znaczące bity wpisywane są do jednego z rejestrów R1, R3 lub R5, a pozostałych osiem bitów — do jednego z rejestrów R0, R2 lub R4, odpowiednio dla kanałów A, B lub C.

Programowanie częstotliwości przebiegu bazowego generatora szumów odbywa się przez zapisanie pięciu mniej znaczących bitów rejestru R6.

Mikser programuje się wykorzystując rejestr R7. Wpisanie zera na bicie B0, B1 lub B2 otwiera przejście dla tonu podstawowego — odpowiednio dla kanału A, B lub C. Wpisanie zera na bicie B3, B4 lub B5 analogiczne otwiera przejście szumu.

Wpisanie zera na bicie B6 rejestru R6 definiuje port IOA jako wejście i wreszcie zero wpisane na najbardziej znaczącym bicie (B7) definiuje port IOB jako wejście.

Określony poziom wyjściowy dla poszczególnych kanałów programuje się wpisując odpowiednią wartość na czterech mniej znaczących bitach rejestrów R8, R9 lub R10.

Okres powtarzania obwiedni programowany jest wpisem 16-bitowej wartości do rejestrów R11 i R12 (bardziej znaczący bajt do R12). Kształt obwiedni definiowany jest przez zapis odpowiedniej kombinacji na czterech mniej znaczących bitach rejestru R13 (rys. 4). Bit B0 odpowiada funkcji HOLD, bit B1 funkcji ALTERNATE, bit B2 funkcji ATTACK i bit B3 funkcji CONTINUE.

Oprac. AJP

W poprzedniej części opisaliśmy komputer IBM-PC i przykładowe procedury w języku maszynowym. W tej — bibliotekę procedur w języku C.

Grafika IBM-PC (2)

Zasady korzystania z przerwania graficznego INT 10H można poznać analizując przedstawione na wydruku 3 procedury zapisane w języku C. Wszystkie procedury wywołują podprogram **video (ax, bx, cx, dx)**, dla którego parametrami są wartości odpowiednich rejestrów. Podprogram ustawia wszystkie potrzebne rejestry i wywołuje przerwanie INT 10H (wydruk 1).

```
public video
video:
        push    bp
        mov     bp, sp
        push    di
        mov     ax, (bp+4)      ; ax <— 1 parametr
        mov     bx, (bp+6)      ; bx <— 2 parametr
        mov     cx, (bp+8)      ; cx <— 3 parametr
        mov     dx, (bp+10)     ; dx <— 4 parametr
        int     10h             ; przerwanie graficzne
        pop     di
        pop     bp
        ret
```

Wydruk 1. Procedura video (ax, bx, cx, dx)

UWAGA: Podprogram wideo z przyrostkiem **-ax, -bx, -cx, -dx** jest podobny do video, z tym że wartością powrotną procedury jest wartość odpowiedniego rejestru.

Procedury wykorzystują również inny podprogram napisany w języku wewnętrznym, który mniej znaczące bajty dwóch argumentów łączy w jedną 2-bajtową liczbę całkowitą (wydruk 2).

```
        public con
con:
        push    bp
        mov     bp, sp
        mov     ah, (bp+4)      ; ah <—1.parametr
        mov     al, (bp+6)      ; al<—2.parametr
        pop     bp
        ret
```

Wydruk 2. Procedura con (rh, rl)

```
/*                      BIBLIOTEKA PROCEDUR
                          W JEZYKU "C"

                    OBSLUGA PRZERWANIA:  INT 10H

                                                                    */

/* WYBOR TRYBU PRACY                                    (A/N,GR)
        mode = 0 -  40 x 25  monochromatyczny A/N
             = 1 -  40 x 25  kolorowy A/N
             = 2 -  80 x 25  monochromatyczny A/N
             = 3 -  80 x 25  kolorowy A/N
             = 4 - 320 x 200 kolorowy GR
             = 5 - 320 x 200 monochromatyczny GR
             = 6 - 640 x 200 monochromatyczny GR                    */
select_mode (mode)
  int mode;
{ video (con (0,mode),0,0,0); }

/* WYBOR TYPU KURSORA                                   (A/N)
        upp = 0,...,7 - linia poczatkowa
        low = 0,...,7 - linia koncowa                               */
cursor_type (upp,low)
  int upp,low;
{ video (con (1,0),0,con (upp,low),0); }

/* POZYCJONOWANIE KURSORA                               (A/N)
        page = 0,...,7 - numer strony (zobacz : select_page)
        row  = 0,...,24 - numer wiersza
        col  = 0,...,79 - numer kolumny                             */
position (page,row,col)
  int page,row,col;
{ video (con (2,0),con (page,0),0,con (row,col)); }

/* ODCZYT POZYCJI KURSORA                               (A/N)
        page = 0,...,7 - numer strony (zobacz : select_page)
WYJSCIE ah = 0,...,24 - numer wiersza
        al = 0,...,79 - numer kolumny                               */
int read_position (page)
  int page;
{ return (video_dx (con (3,0),con (page,0),0,0)); }

/* WYBOR AKTYWNEJ STRONY OBRAZU                         (A/N,GR)
        page = 0,...,7 - numer strony dla A/N 40x25
             = 0,...,3 - numer strony dla A/N 80x25
             = 0       - numer strony dla GR                        */
select_page (page)
  int page;
{ video (con (5,0),con (page,0),0,0); }

/* OBROT OBSZARU W GORE                                 (A/N)
        lines = 0,...,24 - ilosc linii obrotu
        attr  = 0,...,255 - atrybut koloru dopisywanych linii
        row_u = 0,...,24 - wiersz lewego gornego rogu obszaru
        col_l = 0,...,79 - kolumna lewego gornego rogu obszaru
        row_d = 0,...,24 - wiersz prawego dolnego rogu obszaru
        col_r = 0,...,79 - kolumna prawego dolnego rogu obszaru */
roll_up (lines,attr,row_u,col_l,row_d,col_r)
  int lines,attr,row_u,col_l,row_d,col_r;
{ video (con (6,lines),con (attr,0),con (row_u,col_l),con (row_d,col_r))
; }

/* OBROT OBSZARU W DOL                                  (A/N)
        lines = 0,...,24 - ilosc linii obrotu
        attr  = 0,...,255 - atrybut koloru dopisywanych linii
        row_u = 0,...,24 - wiersz lewego gornego rogu obszaru
        col_l = 0,...,79 - kolumna lewego gornego rogu obszaru
        row_d = 0,...,24 - wiersz prawego dolnego rogu obszaru
        col_r = 0,...,79 - kolumna prawego dolnego rogu obszaru */
roll_dn (lines,attr,row_u,col_l,row_d,col_r)
  int lines,attr,row_u,col_l,row_d,col_r;
{ video (con (7,lines),con (attr,0),con (row_u,col_l),con (row_d,col_r))
; }

/* ODCZYT ZNAKU I ATRYBUTU                              (A/N)
        page = 0,...,7 - numer strony (zobacz : select_page)
WYJSCIE ah = 0,...,255 - atrybut znaku
        al = 0,...,255 - znak                                       */
read_ac (page)
  int page;
{ return (video_ax (con (8,0),con (page,0),0,0)); }

/* ZAPIS ZNAKU I ATRYBUTU                               (A/N)
        chr  = 0,...,255 - znak
        page = 0,...,7   - numer strony (zobacz : select_page)
        attr = 0,...,255 - atrybut znaku
        time = 0,...     - ilosc powtorzen                          */
write_ac (chr,page,attr,time)
  int chr,page,attr,time;
{ video (con (9,chr),con (page,attr),time,0); }
```

```
/* ZAPIS ZNAKU                                    (B/N)
        chr  = 0,...,255 - znak
        page = 0,...,7   - numer strony (zobacz : select_page)
        time = 0,...     - ilosc powtorzen                    */
write (chr,page,time)
  int chr,page,time;
{ video (con (10,chr),con (page,0),time,0); }

/* WYBOR PALETY                                   (GR)
        nr = 0,1 - numer palety                               */
pallette (nr)
  int nr;
{ video (con (11,0),con (1,nr),0,0); }

/* WYBOR KOLORU TLA                               (GR)
        color = 0,...,15 - kolor tla                          */
background (color)
  int color;
{ video (con (11,0),con (0,color),0,0); }

/* RYSOWANIE PUNKTU                               (GR)
        color = 0,...,3 - kolor punktu

        u = 0,...,319 - 1.wspolrzedna
        v = 0,...,199 - 2.wsplorzedna                         */
w_dot (color,u,v)
  int color,u,v;
{ video (con (12,color),0,u,v); }

/* RYSOWANIE PUNKTU Z UWZGLEDNIENIEM POPRZEDNIEGO KOLORU (EXCLUSIVE OR)
                                                  (GR)
        color = 0,...,3 - kolor punktu
        u = 0,...,319 - 1.wspolrzedna
        v = 0,...,199 - 2.wspolrzedna                         */
x_dot (color,u,v)
  int color,u,v;
{ video (con (12,color ! 0x80),0,u,v); }

/* ODCZYT KOLORU PUNKTU                           (GR)
        u = 0,...,319 - 1.wspolrzedna
        v = 0,...,199 - 2.wspolrzedna
WYJSCIE a1 = 0,...,3 - kolor                                  */
r_dot (u,v)
  int u,v;
{ return (video_ax (con (13,0),0,u,v)); }
```

Wydruk 3.

LITERATURA

[1] Kernigham B. W., Ritchie D. M.: The C Programming Language. Prentice-Hall, London, 1983

[2] Lafore R. W.: Assembly Language Primer for the IBM PC & XT. The Wait Book, New York, 1984

[3] Morgan Ch. L.: Bluebook of Assembly Routines for the IBM PC & XT. The Wait Book, New York, 1984

[4] Norton P.: Inside the IBM PC Access to Advanced Feature and Programming. Prentice-Hall, Maryland, 1983

[5] Technical Reference. IBM Library, 1983.

ANTONI URBAN

Computer Studio Kajkowscy
Gdynia

Urząd Pocztowy pewnie jeszcze nie prędko dopuści posiadaczy prywatnych komputerów do linii telefonicznych. Poza „względami na interes państwa", przemawia za tym fatalny stan sieci telefonicznej, która po dołączeniu żarłocznych modemów zapewne zakorkowałaby się na amen. Rozważanie standardowego połączenia komputera z modemem wydaje się być zatem pozbawione głębszego sensu. Jednak ten sam standard może być użytecznie wykorzystany do wielu innych także celów, w naszym zaścianku. Przykładem niech będzie tu możliwość przekazywania plików danych między komputerami, które nie mają ze sobą nic wspólnego; poza — łączem RS 232C.

Standard RS 232C

Istnieje kilka standardów określających rozwiązanie interfejsów szeregowych. Jednym z najbardziej popularnych jest RS 232C i bardzo do niego zbliżony V24.

Standard RS 232C określony został przez EIA (Electronic Industries Association) jako opis interfejsu umożliwiającego współpracę terminala nadawczo-odbiorczego z modemem.

Terminal to urządzenie pozwalające użytkownikowi na wprowadzenie informacji (komendy, dane itp.) i odbieranie informacji (np. na ekranie monitora lub wydrukowanej na papierze). Modem (modulator — demodulator) to urządzenie pośredniczące, umożliwiające dołączenie się do sieci telekomunikacyjnej.

Dla określenia źródeł generowanych sygnałów w przypadku współpracy terminal-modem przyjęto następujące oznaczenia:
DCE (ang. Data Communication Equipment) — modem
DTE (ang. Data Terminal Equipment) — terminal nadawczo-odbiorczy.

Z chwilą upowszechnienia mini- i mikrokomputerów, transmisja szeregowa okazała się wygodną metodą realizacji współpracy różnych urządzeń. Chociaż istnieje wiele różnorodnych rozwiązań, nadal często wykorzystuje się standard RS 232C (V24), co ułatwia łączenie sprzętu dostarczonego przez różnych producentów.

Eksplozja komputerów osobistych spowodowała dalszy rozwój sieci telekomunikacyjnych. Bardzo często do komputera domowego dokupywany jest modem i użytkownik uzyskuje — przez przyłączenie do linii telefonicznej — dostęp do plików informacji obsługiwanych przez duże systemy. Upowszechnieniu uległy też sieci lokalne łączące mikrokomputery znajdujące się na terenie jednej instytucji. Dołączenie do adaptera sieci lokalnej często wykonywane bywa przez interfejs szeregowy w standardzie RS 232C. Stosunkowo rzadkie (choć ciągle jeszcze popularne w sprzęcie profesjonalnym krajowej produkcji) są rozwiązania, w których mikrokomputer łączy się przez RS 232C z terminalem nadawczo-odbiorczym.

Przykładem „domowego" wykorzystania standardu RS 232C może być połączenie dwóch komputerów w celu wymiany danych lub nawet oprogramowania (gdy przechowywane jest ono na różnych nośnikach pamięci masowej).

W dalszej części omówiony zostanie standard RS 232C tak, jak został zdefiniowany, czyli do komunikacji terminal-modem. Warto jednak podkreślić, że mikrokomputer może być traktowany jako tzw. inteligentny terminal (czyli DTE) i źródła sygnałów pozostają bez zmian w stosunku do określonych w standardzie.

W standardzie określono cztery typy linii połączeniowych:

— linie transmisji danych

— linie sterująco-kontrolne

— linie synchronizacji czasowej

— linie masy sygnałowej i ekranu.

Linie sterowane są sygnałami napięciowymi zmieniającymi się w zakresie od —12 do +12 V. Na linii danych stan określany jest jako logiczne „1", gdy sterowana jest ona napięciem ujemnym (ang. Marking State). Gdy linia danych sterowana jest napięciem dodatnim określane jest to jako stan logiczny „0" (ang. Spacing State).

Dla linii kontrolno-sterujących stan aktywny (ang. On) oznacza sterowanie napięciem dodatnim.

Stosowanie napięć sterujących z zakresu od —12 do +12 jest dość niewygodne we współczesnych systemach mikrokomputerowych, gdzie z reguły wykorzystywane bywa tylko jedno napięcie zasilające: +5 V. Dlatego też czasami producenci popularnego sprzętu (np. firma COMMODORE w mikrokomputerze C64) nie realizują standardu „do końca", lecz zostawiają użytkownika z sygnałami o poziomach TTL. Jeżeli łączy się dwa komputery leżące obok siebie, to faktycznie konwertery napięć można pominąć. Jeśli jednak chcemy przyłączyć mikrokomputer C64 do modemu, trzeba dokupić dodatkową płytkę, która „dokańcza" realizację standardu.

W tabeli zestawiono sygnały przewidziane w standardzie RS 232C. Zaznaczono również sygnały spoza standardu, wprowadzone w sprzęcie krajowej produkcji.

Sygnały interfejsu RC 232C

EIA RS 232C	CCITT V24	Wyprowadzenia	Oznaczenie mnemoniczne	Źródło	Nazwa linii interfejsu
AA	101	2	—	—	ekran
AB	102	7	GND	—	masa sygnałowa
AB	103	2	TxD	DTE	dane nadawane (ang. Transmitted Data)
BB	104	3	RxD	DCE	dane odbierane (ang. Recived Data)
CA	105	4	RTS	DTE	żądanie nadawania (ang. Request to Send)
CB	106	5	CTS	DCE	gotowość nadawania (ang. Clear to Send)
CC	107	6	DSR	DCE	gotowość modemu (ang. Data Set Ready)
CD	108.2	20	DTR	DTE	żądanie dołączenia modemu do linii (ang. Data Terminal Ready)
CE	125	22	—	DCE	dzwonek (ang. Ring Indicator)
CF	109	8	DCD	DCE	nośna odbierana (ang. Data Carrier Detect)
CG	110	21	SQD	DCE	poziom sygnału odbieranego (ang. Signal Quality Detector)
—	—	21	—	DTE*)	zgłoszenie zapełnienia bufora DZM 180 KSR
CH	111	23	—	DTE	wybór prędkości transmisji (ang. Data Signal Rate Selector)
CI	112	23	—	DCE	
DA	113	24	—	DTE	zegar transmisji nadajnika (ang. Transmiter Signal Element Timing)
DB	114	15	—	DCE	zegar transmisji nadajnika (ang. Transmiter Signal Element Timing)
DD	115	17	—	DCE	zegar transmisji odbiornika (ang. Reciver Signal Element Timing)
SBA	118	15	—	DTE	dane nadawane — kanał 2 (ang. Secondary Transmitted Data)
—	—	14		DTE*)	zgłoszenie zajętości monitora MERA 7952
SBB	119	14	—	DCE	dane odbierane — kanał 2 (ang. Secondary Recived Data)
SCA	120	19	—	DTE	żądanie nadawania — kanał 2 (ang. Secondary Request to Send)
SCB	121	13	—	DCE	gotowość nadawania — kanał 2 (ang. Secondary Clear to Send)
SCF	122	12	—	DCE	nośna odbierana — kanał 2 (ang. Secondary Data Carrier Detect)

*) Sygnały niestandardowe

Standard dopuszcza możliwość zredukowania niektórych linii kontrolnych i linii synchronizacji czasowej. W minimalnych zestawach wykorzystywane są tylko linie transmisji danych **TxD** i **RxD** (i oczywiście linia masy). W wersjach uproszczonych najczęściej wykorzystywane są linie **RTS, CTS, DSR** i **DTR.**

W zasadzie, aby mogła odbywać się transmisja po liniach danych, wszystkie wymienione linie sterujące powinny być w stanie aktywnym. Sesja współpracy zaczyna się od wymuszenia przez terminal (komputer) stanu aktywnego na linii **DTR.** Urządzenie komunikacyjne (modem) powinno potwierdzić gotowość do współpracy sygnałem aktywności na linii DSR [1]).

Komunikacja inicjowana jest przez terminal podaniem stanu aktywnego na linię RTS. Jeżeli modem przyłączony jest do linii, to odpowiada sygnałem CTS. Od tego momentu terminal może wysyłać dane po linii TxD. Dane wprowadzane do terminala (komputera) po linii RxD są ważne, gdy aktywny jest sygnał DCD i SQD.

Gdy łączymy mikrokomputer z terminalem, powstaje nieco zamieszania. Przyjmuje się wtedy, że w układzie terminal (DTE) — modem (DCE) komputer przyłącza się w miejsce modemu. Ponieważ łącze w komputerze przystosowane jest do współpracy z modemem, niezbędne jest zastosowanie specjalnego kabla przejściowego. W kablu tym należy przede wszystkim zamienić (na jednej z łączówek) linie TxD i RxD. Należy również zamienić miejscami linie DSR i DTR. Natomiast linie RTS i CTS najlepiej zapętlić, osobno dla każdego z urządzeń. Ponadto sygnały z linii RTS należy podłączyć do wejść DCD i SQD współpracującego urządzenia. Dodatkowy problem może wyniknąć, gdy któreś z urządzeń nie generuje sygnału DTR. Należy wtedy w drugim urządzeniu zapętlić DTR i DSR.

Sytuacja — gdy łączymy ze sobą dwa komputery — jest bardzo zbliżona do sytuacji łączenia komputera i terminala. Podobnie jak poprzednio, jeden z komputerów musi „wczuć się" w rolę modemu, podczas gdy drugi spełnia funkcję inteligentnego terminala. Uwagi dotyczące przełączania lub zapętlania niektórych linii można odnieść i do tej sytuacji.

Standard nie określa postaci przesyłanych danych. Nie narzuca też, czy transmisja ma być synchroniczna czy asynchroniczna. Najczęściej stosowana jest jednak transmisja asynchroniczna. Przesyłana dana rozpoczyna się od tzw. bitu startowego. Inicjuje on próbkowanie przez odbiornik stanu linii przesyłowej. Jeśli transmisji nie towarzyszy opisany wcześniej protokół wymiany sygnałów sterujących, może się zdarzyć, że transmisja rozpocznie się przed włączeniem odbiornika. Oprócz utraty części informacji, mogą wtedy nastąpić dalsze przekłamania — gdy któryś z bitów informacyjnych zostanie potraktowany jako bit startowy. Jeżeli więc — na przykład — realizujemy przesyłanie między dwoma komputerami, należy zadbać, aby program obsługujący odbiornik rozpoczął działanie przed programem obsługującym nadajnik.

Po bicie startu przesyłanych jest siedem lub osiem bitów informacyjnych, przy czym jako pierwszy wysyłany jest najmniej znaczący bit. Po bitach informacyjnych przesyłany jest bit parzystości, za którym następują bity stopu, rozdzielające informacje. Liczba bitów informacyjnych, rodzaj kontroli (parzystość, nieparzystość lub brak bitu kontrolnego) oraz liczba bitów stopu (1, $1^1/_2$ lub 2) — to parametry transmisji, które należy „uzgodnić" między nadajnikiem i odbiornikiem. Uzgodnić należy też szybkość transmisji. Mierzona jest ona w bodach (bitach na sekundę) i przyjmuje jedną z następujących wartości: 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19 200.

Im dłuższe jest połączenie między odbiornikiem a nadajnikiem, tym mniejszą szybkość transmisji należy stosować. Wskazane też jest stosowanie mniejszych szybkości, gdy nadajniki są realizowane na elementach dyskretnych.

AJP

*) Nazwa DSR (ang. Data Set Ready) jest nieco myląca, gdyż modem wcale nie musi w momencie zgłoszenia dysponować plikiem danych, które przeznaczone są do wyprowadzenia na terminal

prowadzi
ANDRZEJ J. PIOTROWSKI
tel. dom.: 48-22-85